Note: This tutorial assumes basic familiarity with the ABB robot controller. Consult the manufacturer's documentation for further details on any of the steps listed in this tutorial.

Please ask about problems and questions regarding this tutorial on answers.ros.org (http://answers.ros.org). Don't forget to include in your question the link to this page, the versions of your OS & ROS, and also add appropriate tags.

Installing the ABB ROS Server

Description: This tutorial walks through the steps of installing the ROS server code on the ABB robot controller and configuring the required controller settings.

Keywords: ABB, Industrial, IRC5

Tutorial Level: INTERMEDIATE

Next Tutorial: Next, you'll learn how to run and manage the ABB ROS Server Running the server (/abb_driver/Tutorials/RunServer)

Contents

This tutorial has two parts: installing server code and configuring the controller with the appropriate tasks.

1. Prerequisites

The ABB ROS Server code is written in RAPID, using a socket interface and multiple parallel tasks. The code was originally tested on an IRC-5 controller, with an IRB-2400 robot, but other combinations should also work. The following controller options are required:

- `623-1`: *Multitasking*
- `672-1`: *Socket Messaging* (in recent RobotWare versions, this option is included with `616-1`: *PC Interface*)

RobotWare OS version 5.13 or later is required due to the use of certain socket options. Earlier versions may work, but will require modifications to the RAPID code.

2. Installing Server Code

All files in the abb_driver/rapid (https://github.com/ros-industrial/abb_driver) (Indigo and later) directory should be copied to the robot controller. This tutorial assumes the files are copied to a "`ROS`" subdirectory under the system's `HOME` directory (e.g. `/<system>/HOME/ROS/*`).

See the manufacturer's documentation for file-transfer methods. RobotStudio Online and USB drives are often convenient methods to transfer files to the controller.

3. Configuring Controller Settings

The ROS Server code requires 3 tasks. Some modules are loaded to specific tasks, and others are shared between tasks, as described below:

3.1 File Overview

- Shared by all tasks
  - `ROS_common.sys` -- Global variables and data types shared by all files
  - `ROS_socket.sys` -- Socket handling and simple_message (/simple_message) implementation
  - `ROS_messages.sys` -- Implementation of specific message types
- Specific task modules
  - `ROS_stateServer.mod` -- Broadcast joint position and state data
  - `ROS_motionServer.mod` -- Receive robot motion commands
  - `ROS_motion.mod` -- Issues motion commands to the robot

3.2 Create Tasks

1. Browse to *Controller tab* → *Configuration Editor* → *Controller* → *Task*, then right-click *New Task* (In RobotStudio 5, this is found under *ABB* → *Control Panel* → *Configuration* → *Topics* → *Controller* → *Task*)
2. Create 3 tasks as follows:

Name	Type	Trust Level	Entry	Motion Task
ROS_StateServer	SEMISTATIC	NoSafety	main	NO
ROS_MotionServer	SEMISTATIC	SysStop	main	NO
T_ROB1	NORMAL		main	YES

It is easiest to wait until all configuration tasks are completed before rebooting the controller.

NOTES:

1. *The `T_ROB1` motion task probably already exists on your controller.*
2. *If `T_ROB1` has existing motion-control modules, you may need to rename the `main()` routine in `ROS_Motion.mod` to `ROS_main()`. In this case, set the Entry point for `T_ROB1` task to `ROS_main()`.*
3. *For multi-robot controllers, specify the desired robot (e.g. `rob1`) for each task*

4. *`SEMISTATIC` tasks will auto-start when controller is booted. They are visible, but cannot be easily seen for troubleshooting. For debug or development purposes, it may be desired to set both `ROS_*Server` tasks to `Type=NORMAL`.*

3.3 Create Signals

1. Browse to *Controller tab* → *Configuration Editor* → *I/O System* → *Signal*, then right-click *New Signal*

   (In RobotStudio 5, this is found under *ABB* → *Control Panel* → *Configuration* → *Topics* → *I/O* → *Signal*)

2. Create 7 signals as follows:

Name	**Type of Signal**
signalExecutionError	Digital Output
signalMotionPossible	Digital Output
signalMotorOn	Digital Output
signalRobotActive	Digital Output
signalRobotEStop	Digital Output
signalRobotNotMoving	Digital Output
signalRosMotionTaskExecuting	Digital Output

It is easiest to wait until all configuration tasks are completed before rebooting the controller.

3.4 Tie Signals to the System Outputs

1. Browse to *Controller tab* → *Configuration Editor* → *I/O System* → *System Output*, then right-click *New System Output*

   (In RobotStudio 5, this is found under the *ABB* → *Control Panel* → *Configuration* → *Topics* → *I/O* → *System Output*)

2. Add one entry for signal as follows:

Signal Name	**Status**	**Arg 1**	**Arg 2**	**Arg 3**	**Arg 4**
signalExecutionError	Execution Error	N/A	T_ROB1	N/A	N/A
signalMotionPossible	Runchain OK	N/A	N/A	N/A	N/A
signalMotorOn	Motors On State	N/A	N/A	N/A	N/A
signalRobotActive	Mechanical Unit Active	ROB_1	N/A	N/A	N/A
signalRobotEStop	Emergency Stop	N/A	N/A	N/A	N/A
signalRobotNotMoving	Mechanical Unit Not Moving	ROB_1	N/A	N/A	N/A
signalRosMotionTaskExecuting	Task Executing	N/A	T_ROB1	N/A	N/A

3.5 Load Modules to Tasks

1. Browse to *Controller tab* → *Configuration Editor* → *Controller* → *Automatic Loading of Modules*, then right-click *New Automatic Loading of Modules*

   (In RobotStudio 5, this is found under *ABB* → *Control Panel* → *Configuration* → *Topics* → *Controller* → *Automatic Loading of Modules*)

2. Add one entry for each server file as follows:

File	Task	Installed	All Tasks	Hidden
HOME:/ROS/ROS_common.sys		NO	YES	NO
HOME:/ROS/ROS_socket.sys		NO	YES	NO
HOME:/ROS/ROS_messages.sys		NO	YES	NO
HOME:/ROS/ROS_stateServer.mod	ROS_StateServer	NO	NO	NO
HOME:/ROS/ROS_motionServer.mod	ROS_MotionServer	NO	NO	NO
HOME:/ROS/ROS_motion.mod	T_ROB1	NO	NO	NO

3. After the last change, select YES to restart the controller and apply the changes.

3.6 Updating Software

To update robot-server files with new code versions, use the following procedure to ensure that the changes are actually applied:

1. Copy the new/updated files onto the robot controller, as before.
2. Restart the controller using a **P-Start**
   - *ABB* → *Restart* → *Advanced* → *P-Start* → *OK*
   - *NOTE: This will erase any existing modules that have been loaded to memory. This may cause compilation issues on restart. If this is a concern, try another method: Warm Start, manually reloading modules (may require setting SEMISTATIC tasks to NORMAL tasks), etc.*
3. After the controller reboots, the new changes should be active.

Wiki: abb_driver/Tutorials/InstallServer (last edited 2020-08-11 12:49:45 by GvdHoorn (/GvdHoorn))